Remarks

Entry of this Amendment, reconsideration of the application and allowance of all claims are respectfully requested. Claims 1, 3, 5, 8-12, 14-18 & 20-24 remain pending.

By this paper, independent claims 1, 3 & 5 are amended to more clearly point out and distinctly claim certain aspects of the present invention. Although Applicants believe that certain of these aspects were previously presented, the claim amendments are submitted in a *bona fide* attempt to further prosecution of this application. Support for the amended language can be found throughout the application as filed. For example, reference pages 5, 11, 14 & 15 of the specification, as well as the subject matter of canceled claims 7, 13 & 19. No new matter is added to the application by any amendment presented.

In the Office Action, all pending claims (i.e., claims 1, 3, 5 & 7-24) were rejected under 35 U.S.C. §102(e) as being anticipated by Cramer et al. (U.S. Patent No. 5,946,685; hereinafter Cramer). This rejection is respectfully, but most strenuously, traversed to any extent deemed applicable to the claims presented herewith and reconsideration thereof is requested.

As explained by Applicants at pages 2 & 3 of the specification, access management to a global repository typically involves obtaining locks (either shared or exclusive) on requested resources of the repository. Specifically, a thread of an application that wishes to access one or more resources of the repository obtains locks on those requested resources. These locks are specifically identified and attached with the requesting thread. Thus, it is necessary to tap into and understand the threading model of the operating system. This creates complications and makes the locking facilities platform dependent.

This problem is addressed by Applicants, in one aspect, through a method of managing locking of resources of a global data repository of a distributed computing environment (e.g., claim 1). The method includes: issuing a request, via a thread of a multithreaded client application of the distributed computing environment, for a lock of a resource associated with a server data tree of the global data repository; and obtaining the lock for the multithreaded client application. Because the lock is obtained for the multithreaded client application, the lock is independent of a threading model of a requesting thread of the multithreaded client application. The obtaining of the lock includes employing a local tree which is local to the client application

and has a mount point usable by the client application to lock the resource via the server data tree. The local tree is a data tree accessible by a plurality of threads of the multithreaded client application. Further, the resource is also lockable via another mount point of either a local tree or another local tree.

Advantageously, employing local trees (each of which is local to a client application) allows client applications to access a resource (e.g., a table) via a mount point (e.g., directory) of a local tree even though the resource is locked in a global data repository (e.g., global data space). Further, the same resource (table) can be mounted and locked through different mount points (directories). For instance, with a global database that includes Table X, and a local tree that includes directories A and B, an application can mount and lock Table X through directory A and mount and lock Table X through directory B. Therefore, in this example, Table X is mounted and locked more than once, as if it were two different resources.

It is well-settled that there is no anticipation of a claim unless a single prior art reference discloses: (1) all the same elements of the claimed invention; (2) found in the same situation as the claimed invention; (3) united in the same way as the claimed invention; and (4) in order to perform the identical function as the claimed invention. In this instance, Cramer fails to disclose certain aspects of Applicants' invention as recited in the independent claims 1, 3 & 5, and as a result, does not anticipate (or even render obvious) Applicants' invention.

Cramer describes a global mount mechanism capable of maintaining a consistent global name space in a distributed computing system including a plurality of nodes interconnected by communications links. The global mount mechanism mounts a new file system resource into the global name space in a coherent manner such that the new file system resource is mounted at the same mount point concurrently in each node. The global mount mechanism accommodates mount or unmount requests initiated from a requesting node for a resource located in a remote node. The global mount mechanism is also used to unmount a file system resource from the global name space. The global mount mechanism includes an initialization procedure that is used to generate the global name space initially by providing each local mount point with a global locking capability. (See Abstract.)

In their independent claims, Applicants recite a particular technique for managing locking of resources of a global data repository. This technique includes obtaining a lock of a resource

associated with a server data tree of a global data repository for a multithreaded client application. As expressly recited by Applicants, this lock is independent of a threading model of the requesting thread of the multithreaded client application. In Applicants' technique, the thread requests the lock, but the lock is obtained by the multithreaded client application, rather than the thread itself as in the conventional approach.

With respect to this aspect, Applicants submit that a careful reading of Cramer fails to uncover any teaching or suggestion of a lock on a resource being obtained by a multithreaded client application *per se*. Further, a careful reading of Cramer fails to uncover any discussion of a lock being independent of a threading model of the requesting thread of the multithreaded client application.

With respect to this aspect of their invention, the Office Action references column 2, lines 26-38 and column 4, lines 11-15 of Cramer. At these lines, Cramer describes a global mount mechanism wherein each node has a local name space including a number of local file system resources that are only accessible from within the node. Cramer's initialization mechanism provides one or more local directories or local mount points with a global locking capability that enables the local mount to be locked by any node in the cluster. The global locking capability turns the local mount point into a global mount point that is part of the global name space. One or more local file system resources can then be mounted at a global mount point and, hence, become part of the global name space. Applicants respectfully submit that this teaching of Cramer is not relevant to their recited technique.

For example, Applicants recite issuing a request, via a thread of a multithreaded client application of the distributed computing environment, for a lock of a resource associated with a server data tree of the global data repository, and based upon this request, a lock is obtained for the multithreaded client application. This lock is recited to be independent of a threading model of the requesting thread of the multithreaded client application, i.e., the lock is owned by the multithreaded client application, rather than the thread requesting the lock. No similar functionality is taught or suggested by Cramer. To the extent that Cramer's global mount mechanism employs a lock, it is noted at column 2, line 33 that the local mount may be locked by any node in the cluster. Thus, Cramer is describing a lock that is owned by a node of the cluster, and not by a multithreaded client application *per se*. Further, there is simply no teaching

or suggestion that the lock obtained in Cramer is in any way independent of a threading model of the requesting thread of the multithreaded client application. For at least these reasons, Applicants respectfully request reconsideration and withdrawal of the anticipation rejection to the independent claims presented.

Still further, Applicants' independent claims recite that the local tree at issue is local to the client application. In comparison, the local tree in Fig. 2A of Cramer is owned by a node of the cluster, i.e., the operating system running on the node. This is clearly distinct from the process of Applicants' invention.

Applicants further recite that the local tree (which is local to the client application) is a data tree accessible by the plurality of threads of the multithreaded client application. Again, the local tree in Cramer is associated with the node, as shown in Fig. 2A thereof. In addition, Cramer describes a distributed locking mechanism by teaching that the local mount point can be turned into a global mount point, thus locking from one client to another client. This functionality is simply not applicable to the management process recited by Applicants. In Applicants' approach, a local tree of the multithreaded client application is employed in obtaining the lock for the multithreaded client application. This local tree is a data tree accessible by a plurality of threads of the multithreaded client application. Thus, in Applicants' process, the lock resource can be accessed by multiple threads of the multithreaded client application. No similar functionality is provided by Cramer, or the other art of record.

For at least the above reasons, Applicants respectfully submit that Cramer does not anticipate their invention as recited in the claims at issue. A careful reading of Cramer fails to uncover any teaching or suggestion that a lock is obtained for a multithreaded client application *per se*, let alone that the lock is independent of a threading model of the requesting thread of the multithreaded client application. Still further, there is no teaching or suggestion in the applied art for employing a tree local to the multithreaded client application in obtaining the lock, wherein the tree is a data tree accessible by a plurality of threads of the multithreaded client application. Yet further, there is no teaching or suggestion in Cramer that the locked resource is further lockable via another mount point of the local tree of the client application or another local tree.

Based upon the above, reconsideration and withdrawal of the anticipation rejection to the independent claims is respectfully requested. The dependent claims are believed allowable for the same reasons as the independent claims, as well as for their own additional characterizations.

Applicants respectfully submit that all claims are in condition for allowance, and such action is respectfully requested.

If a telephone conference would be of assistance in advancing prosecution of the subject application, Applicants' undersigned attorney invites the Examiner to telephone him at the number provided.

Respectfully submitted,

Kevin P. Radigan

Kevin P. Radigan
Attorney for Applicants
Registration No.: 31,789

Dated: March 03, 2005.

HESLIN ROTHENBERG FARLEY & MESITI P.C.
5 Columbia Circle
Albany, New York 12203-5160
Telephone: (518) 452-5600
Facsimile: (518) 452-5579